

**Supporting online material
for:
LocTree2 predicts localization for all domains of life**

Tatyana Goldberg, Tobias Hamp & Burkhard Rost

SOM Section 1: LocTree2 development workflow

We extracted sets of archaeal, bacterial and eukaryotic proteins together with their experimentally determined annotations of subcellular localization from the SWISS-PROT database (Bairoch and Apweiler, 2000) (Methods). We internally homology reduced these data sets to avoid homology-based inference of subcellular localization (Methods). We built three separate classification models, one for each domain of life (Fig. 1). This means we carried out the entire following procedure three times.

As a first step, we divided a data set into five equally sized subsets (Methods) in order to train and test various classification models via cross-validation. Applying “stratification”, we made sure that classes had about the same size in each of the five sets. There was nothing special in the way we carried out the cross-validation: in one fold, four subsets were used for training and one for testing. Then, all subsets were rotated such that each subset was used for testing exactly once. Finally, we averaged the performances over all test sets. We always introduced a certain degree of homology within each of the training sets in order to increase them in size (SOM Section 4). This was found to be beneficial and did not compromise the similarity between training and test proteins.

In this work, we essentially wanted to study the power of support vector machines (SVMs) (Cortes and Vapnik, 1995) in combination with string kernels for subcellular localization prediction. A kernel is the most crucial part of a SVM: it determines the ‘feature space’, i.e. the space into which the objects under consideration (here: protein sequences) are mapped and where they are linearly separated. The better the kernel, the better we will be able to discriminate between proteins from different localization classes. As we have to apply a kernel function during the developmental phase many times for training and testing, its speed also majorly determines the degree with which we can optimize, e.g. free parameters and how fast new, unseen proteins can be classified.

In our case, the problem of optimizing free parameters was ubiquitous: it started with the choice of the multi-class classification scheme and the underlying kernel function. Each kernel, in turn, had at least two other parameters. Additionally, we had to optimize the SVM complexity parameter C and perform Platt-Scaling (Platt, 1999) for each binary SVM (Platt Scaling converts SVM scores into probabilities). Because optimizing all these parameters at once would have created impossible amounts of value combinations and learning tasks, we decided to solve the problem in three major steps. In other words, given parameters A, B, C, D this essentially meant that we first optimized parameters A, B , and then C, D with the best values for A, B . We understand that this does not find a ‘global optimum’ in the sense of the best parameter combination. As we later show, however, the ‘local optimum’ that we find is better than most, if not all, other current subcellular localization predictors. Besides, we always put great emphasis on not allowing test data to ‘leak’ into the training data, what sometimes led to quite complicated setups. The WEKA package (Holmes, et al., 1994) was an invaluable help in this process and we provide command line calls of each step upon request.

In our first step, we compared the performances of three different string-based kernel functions, namely the String Subsequence Kernel (Lodhi, et al., 2002), the Mismatch Kernel (Leslie, et al., 2004) and the Profile Kernel (Kuang, et al., 2004; SOM Section 2). At this early stage of our evaluation, we decided to only use the simplest and fastest multi-class classification approach one-against-all (Allwein and Singer, 2000). We performed one entire cross-validation (above) for each kernel and then chose the winner for subsequent steps. In each cross-validation fold, we optimized the different kernel-specific parameters.

We optimized kernel specific parameters in a nested ten-fold cross-validation: an 'outer' training set (comprising 4/5 of the original data set; before) was again split into 10 'inner' partitions. We then trained a one-against-all model with a particular parameter combination on 9 of these splits and tested it on the remaining one. This was repeated ten times by rotating through the 'inner' folds and we obtained the average performance for this parameter-combination and 'outer' training set. Then, we changed the parameter combination and repeated the nested cross-validation. Finally having calculated the performance for all parameter combinations, we chose the one leading to the best performance. This combination was then used to train a model using all 10 inner splits and to predict proteins in the outer test split. We repeated this procedure five times by rotating through the outer cross-validation folds in order to predict all proteins. As mentioned before, we chose the kernel leading to the highest overall accuracy as the winner. This was the Profile Kernel. Depending on the kingdom (archaea, bacteria or eukaryota), it was found to be either within the standard errors of other kernel functions or significantly more accurate. Additionally, it was much faster in runtime compared to other kernel functions (data not shown).

Having found the best kernel in this way, we assessed in our second step the performance of different multi-class classification approaches. These were One-Against-All (Allwein and Singer, 2000), Ensembles of Nested Dichotomies (Frank and Kramer, 2004), Ensembles of Class Balanced Dichotomies (Dong, et al., 2005), Ensembles of Data Balanced Dichotomies (Dong, et al., 2005) and Nested Dichotomies of a Fixed Structure (Methods; Fig. 1). We briefly introduce them in SOM Section 3. We again evaluated the performance for each of the five splits of the training set in a stratified 10-fold cross-validation. The only difference to before was that instead of three different kernels and one multi-class approach, we now evaluated one kernel (the Profile Kernel) and five multi-class classification schemes.

We observed a superiority of dichotomies-based classification approaches over one-against-all, but could not find a significant difference in prediction performance among them. However, measuring the classification speed (the number of protein sequences processed per minute) revealed a significant advantage of nested dichotomies with a fixed structure (Fig. 1; Table SOM_3) and we chose these models for our final step (note that each dichotomy is different for the three kingdoms because they have different localization classes).

So far, we have found the optimal kernel (the Profile Kernel) and the multi-class classification scheme (Nested Dichotomies of a Fixed Structure). We have still neglected the optimization of the SVM cost parameter C and have not used Platt Scaling. In our third and last step, we included them in the form of two additional 5-fold cross-validation layers for each binary SVM (increasing the overall number of layers to 4). For example, in the second outer cross-validation fold, the seventh inner fold, the Profile Kernel parameter combination ($k=2$ and $\sigma=6$), and the root node of the nested dichotomy of a fixed structure for archaea, we have a binary data set for the corresponding SVM (cytoplasmic vs. non-cytoplasmic). In order to find its best parameter C , we performed a 5-fold cross-validation for each possible value (0.01, 0.1, 1.0, 10, 100, 1000). For each of these $5 \times 6 = 30$ folds, we performed Platt-Scaling, using a 5-fold cross-validation again.

In Fig. 2, we present the average performance over the five outer folds of our model after this last step.

The final models (one for each kingdom), which we evaluated against current state-of-the-art prediction methods for subcellular localization (Methods), and that we installed on our server, were obtained after a final re-training using all of the five outer folds as training data. The Profile Kernel parameters for our final models were: $k=3$ and $\sigma=5$ for archaea, $k=5$ and $\sigma=9$ for bacteria, $k=6$ and $\sigma=11$ for eukaryota.

An in-depth analysis of these models with respect to unknown signal peptides is theoretically possible, but necessary methodologies are yet to be developed. Nevertheless, we provide a preliminary analysis of one of our SVMs in SOM Section 5.

SOM Section 2: The Profile Kernel.

There are a number of sequence-based kernel functions designed for protein classification tasks. In this work, we applied and compared the String Subsequence Kernel (Lodhi, et al., 2002), the Mismatch Kernel (Leslie, et al., 2004) and the Profile Kernel (Kuang, et al., 2004). The main idea behind them is to compare two protein sequences by looking at the number of common subsequences of a fixed length. No biological knowledge is incorporated, in the sense that protein sequences are simply represented as strings of amino acids. In the following, we introduce the main principles behind the Profile Kernel, the kernel function selected to be used for the LocTree2 classification system.

Evolutionary sequence profile. The key feature of the Profile Kernel, as the name already states, is the use of protein sequence profiles. Such a profile is estimated by aligning a target sequence against a group of homologous (similar) sequences, e.g. obtained via BLAST (Altschul, et al., 1990), and compiling the conservation of each amino acid at each alignment position into a score. Many different ways have been proposed on how to compute these scores, with the most popular variant arguably being the one by Altschul et al. and implemented in PSI-BLAST (Altschul et al., 1997). In the profile kernel, the score is simply the negative logarithm of the amino acid frequency at a particular position, slightly 'smoothed out' by pseudo counts (pseudo amino acid probabilities estimated from the training data [Kuang, et al., 2004]). Consequently, in the $n \times 20$ scoring matrix (n being the length of the target sequence and 20 the size of the amino acid alphabet), a value around 0.0 indicates that the respective amino acid has often been observed at a particular position, whereas higher values mean the opposite, i.e. weak or no conservation. We obtained position specific frequency matrices from PSI-BLAST by querying the target sequence against a redundancy reduced combination of SWISS-PROT, TrEMBL (Bairoch and Apweiler, 2000) and PDB (Berman, et al., 2000) (Methods).

Computation of the Profile Kernel. The Profile Kernel makes use of an evolutionary profile P_s of a sequence s . The user has to define the length of the subsequences to consider (k) and the conservation threshold σ . The latter defines a filter for k -mers which exhibit high sequence diversity.

More formally: Given k -mer $m(s, j)$,

$$m(s, j) = s[j+1 : j+k] = s_{j+1} \dots s_{j+k} \text{ with } 0 \leq j \leq |s| - k$$

The kernel looks at the corresponding part of the profile ($P_{(s, j)}$, i.e. the profile P_s reduced to substitution scores between residues $j+1$ and $j+k$) and determines all k -mers with a cumulative substitution score below σ :

$$S_j = \left\{ x \mid x \in \Sigma^k \wedge - \sum_{i=1}^k \log P_{(s, j)}(i, x_i) < \sigma \right\}$$

with Σ being the alphabet of 20 amino acids and $P_{(s, j)}(i, x_i)$ the frequency of amino acid x_i at position i in the sub-profile $P_{(s, j)}$.

Then, we can define the feature map of the kernel:

$$\Phi^x(P_s) = \sum_{j=0}^{|s|-k} I(x \in S_j)$$

$$\text{with } x \in \Sigma^k,$$

$\Phi^x(P_s)$ indicating the value of k -mer x in the feature vector $\Phi(P_s)$ and

$$I(x) = 1 \text{ if } x \in S_j.$$

Note that $\Phi(P_s)$ has $|\Sigma|^k = 20^k$ dimensions. Consequently, the Profile Kernel is defined as the dot product of two feature vectors:

$$k(P_{s1}, P_{s2}) = \Phi(P_{s1}) \cdot \Phi(P_{s2})$$

Directly following the procedure above when implementing the kernel would quickly result in unfeasible runtime and memory requirements. Luckily, we can entirely avoid explicitly mapping a profile into the k -mer feature space and directly compute the kernel. This is commonly known as the ‘kernel trick’. Additionally, we can combine the computation of kernel values and create the entire kernel matrix in one operation. The kernel matrix is an all-against-all comparison of each protein in the data set. Each cell contains the kernel value of the two proteins under comparison. SVMs either create it on their own during training or receive it from the user.

The Profile Kernel applies an efficient data structure that is built on all k -mers, called suffix trie, for the efficient computation of the kernel matrix. k -long profiles are stored on the path from the root to the leaf. An internal node of depth d stores a set of pointers to all k -length profiles $P_{(s,j)}$, whose current cumulative conservation scores are less than the σ threshold.

More specifically:

$$n_p = \left\{ P_{(s,j)} \mid - \sum_{i=1}^d \log P_{(s,j)}(i, s_i) < \sigma \wedge m(s,j)[1:d] = \text{seq}(n) \right\}$$

where n_p is the set of k -long sub-profiles stored at node n and $\text{seq}(n)$ is the sequence induced by the path from the root to node n .

Only processing the profiles remaining at the leaf nodes, we can save a lot of computation and compute many kernel values at the same time. We refer to Leslie et al, 2004 and Kuang et al. 2005 for a more detailed description of the procedure.

Generally, the complexity of computing a Profile Kernel value $k(P_{s1}, P_{s2})$ depends on how many k -mers fall below σ . It has been empirically observed (Kuang, et al., 2005) that with a typical choice of σ , one k -mer in the original sequence translates into $m=1, 2$ slightly different k -mers at the same position. It can then be shown that the worst case complexity of computing a kernel value for a pair of proteins of lengths l_1 and l_2 is $O(k^{m+1}|\Sigma|^m(l_1 + l_2))$. In practice, however, we usually achieve much lower complexity. We again refer to Kuang, et al., 2005 for a more detailed analysis.

SOM Section 3: Multi-class classification schemes

One-Against-All (Allwein and Singer, 2000). Given a set of n classes, n different binary classifiers are employed such that each classifier discriminates between the positive training instances belonging to one class and the negative training instances belonging to the remaining $n-1$ classes. The classification result is the output of the classifier that generates the highest value.

Ensemble of Nested Dichotomies (ENDs) (Frank and Kramer, 2004). A set of twenty randomly composed nested dichotomies (NDs) (Fox, 1997), represented as binary trees. Each internal node of the tree stores one binary classifier and a set of corresponding classes. The root node contains the entire set of classes and learns to separate it into two subsets – a positive and a negative subset. The two successor nodes of the root inherit two subsets and the procedure is repeated until the leaf node is reached. The number of leaf nodes corresponds to the number of localization classes. The result of an END is the average over the estimates obtained from the individual trees.

Ensemble of Class Balanced Nested Dichotomies (ECBNDs) (Dong, et al., 2005). While ENDs sample from a space of all possible tree structures, ECBNDs sample from a space of class-balanced tree structures and built an ensemble of balanced trees. Each internal node in a class-balanced binary tree has two equal-sized subsets to pass to both its successor nodes, which limits the number of possible sets of classes a node can inherit. As a result, the number of possible CBDNs is always smaller than the number of possible NDs.

Ensemble of Data Balanced Nested Dichotomies (EDBNDs) (Dong, et al., 2005). DBNDs are built by randomly assigning classes to two subsets until the number of instances in one of the subsets exceeds half the total amount of instances in the parental node. The two data-balanced subsets are then passed to the successor node. Thus, the heavily populated classes are located high up in the tree structure making the ensemble of possible DBNDs (EDBNDs) biased towards populous classes. However, it has been shown in (Dong, et al., 2005) that the accuracy of EDBNDs is comparable to that of ENDs and ECBNDs on the UCI dataset (Blake and Merz, 1998). This was the reason for investigating this approach on our data.

Nested Dichotomies of a Fixed structure. The knowledge of general pathways of protein sorting was used to design hierarchical trees of a fixed architecture for archaeal, bacterial and eukaryotic proteins (Fig. 1). The difference to ENDs is essentially that we use biological knowledge to define a single ND, instead of randomly creating multiple random NDs and then averaging.

SOM Section 4: Size increase of the training sets.

It has been shown that larger data sets improve SVM performance through increased coverage of the sequence space (Webb and Yu, 2004). Moreover, SVM performance can also be improved through training on sequence redundant sets. Therefore, we allowed a certain degree of homology within each of the training sets of archaeal, bacterial and eukaryotic proteins and thus increased the size of our training data considerably, by almost a factor of 4.

Outline of the algorithm: (1) Start with the homology reduced set and align it against all proteins extracted from SWISS-PROT (Bairoch and Apweiler, 2000) by a pairwise BLAST (Altschul, et al., 1997) (e.g. BLAST2 at $E\text{-value} < 10^4$ in our case); (2) Compile HSSP-values (Rost, 1999; Sander and Schneider, 1991) for each pair of aligned sequences. (3) Find all structural homologs to the sequences in the homology reduced set at $\text{HSSP-value} \leq 60$; (4) Align all sequences found in the previous step against each other by a pairwise BLAST; (5) Find all pairs that are structural homologs at $\text{HSSP-value} \leq 60$; (6) Remove sequences from the previous step that have $\text{HSSP-value} > 0$ to more than one sequence in the homology reduced set.

SOM Section 5: Analysis of k-mers important for endoplasmatic reticulum association.

Although the profile kernel computes dot products implicitly via a kernel trick, the normal vector of the separating hyperplane can be made explicit (Leslie, et al., 2004). This normal vector defines a weight for each k-mer, indicating its contribution to the final classification of a protein: the higher the absolute value, the more the k-mer contributes to the classification of a protein.

However, the biological implications of such a vector are limited (examples given below) and many technical issues would have to be solved before approaching a systematic analysis of our models with respect to new localization signals. (The final class of a protein is the result of many SVMs, each with different normal vectors, to give only one example.)

Nevertheless, for a proof of principle, we computed the explicit normal vector of the SVM separating soluble proteins of the endoplasmatic reticulum (ER; 26 proteins) from those that are secreted or reside in the Golgi apparatus (non-ER; 2318 proteins; Fig. 1). Necessary programs were available in the profile kernel package. Each of the 64 M dimensions of this vector determined the weight of one particular k-mer ($\text{alphabet size}^{\text{k-mer length}} = 20^6 = 64 \text{ M}$; weights ranged from 0.6 to -1.4). We manually analyzed the 100 k-mers with the highest positive weight, i.e. having the highest impact on the classification as ER-associated (weight range: 0.6 – 0.4). A majority came from the C-terminal domain of Calreticulin proteins (e.g. CALR_BOVIN) and only consisted of aspartic and glutamic acids (e.g. EDEDDE, DDEDDE, DEEDEE, ...), rendering the domain very acidic. Their high weights can be explained by frequent occurrence in the 26 ER training proteins and absence in the support vectors of non-ER proteins. Indeed, due to our including slightly homologous proteins in the training set, the Calreticulin family was a little overrepresented among ER proteins (6 proteins). The fact that those k-mers weighed higher than other parts of Calreticulin proteins, however, indicated their particular importance for ER localization. Several experimental studies confirmed this hypothesis (Villamil Giraldo, et al., 2010).

A second striking class of high scoring k-mers consisted of leucin stretches (e.g. LLLLLL, LLLLLA). They could be found in the N-terminal regions of three diverse proteins, namely LDLR chaperone MESD (MESD_HUMAN), GDP-fucose protein O-fucosyltransferase 1 (OFUT1_RAT) and Orexin (OREX_RAT) and are all part of known or putative signal peptides (e.g. [Sakurai, et al., 1999]). However, to our knowledge, exactly this leucin repeat has so far not been identified as a crucial component of the signal and might therefore be a good target for further experimental analyses.

Curiously, the most well-known ER signaling motif, the KDEL retention sequence, was not among the top scoring k-mers despite being largely present in our dataset: 10 of 26 ER proteins ended with the sequence KDEL or HDEL. Further analyses, however, revealed the signal in a different way: There were $2 \cdot 20^3 = 800$ possible k-mers ending with HDEL or KDEL. 760 of them had a positive weight, only 36 were slightly negative (4 had weight 0.0). This illustrates the need for better methods to detect conserved signals and the limits of the current profile kernel: the location of the motif is important for its function; however it can only partially be captured by our feature space. Many k-mers ending with HDEL or KDEL were also present in non-ER proteins, but not necessarily at the C-terminus.

Table SOM_1: Number of proteins in sequence unique data sets used for the development of LocTree2

<i>Localization</i>	<i>Eukaryota</i>	<i>Bacteria</i>	<i>Archaea</i>
Chloroplast	133	-	-
Chloroplast membrane	11	-	-
Cytosol	220	179	41
Endoplasmic reticulum	10	-	-
Endoplasmic reticulum membrane	65	-	-
Extra-cellular space	596	82	5
Fimbrium	-	16	-
Golgi apparatus	3	-	-
Golgi apparatus membrane	17	-	-
Mitochondria	140	-	-
Mitochondria membrane	87	-	-
Nucleus	320	-	-
Nucleus membrane	5	-	-
Outer membrane	-	6	-
Plasma membrane	40	144	13
Periplasm	-	52	-
Peroxisome	6	-	-
Peroxisome membrane	2	-	-
Plastid	14	-	-
Vacuole	3	-	-
Vacuole membrane	10	-	-
	1682	479	59

The table displays the number of sequences per localization in the sequence unique sets of eukaryotic, bacterial and archaeal proteins. We only used experimentally determined subcellular localization annotations from SWISS-PROT release 2011_04 (Methods).

Table SOM_2: Number of proteins in sequence unique independent test sets

<i>Localization</i>	<i>New SWISS-PROT</i>		<i>LocDB</i>	
	<i>Bacteria</i>	<i>Eukaryota</i>	<i>A. thaliana</i>	<i>H. sapiens</i>
Chloroplast	-	-	-	-
Cytosol	10	5	2	58
Endoplasmic reticulum (ER)	-	-	3	8
ER membrane	-	3	-	1
Extra-cellular space	8	15	8	14
Fimbrium	1	-	-	-
Golgi apparatus	-	-	1	6
Golgi apparatus membrane	-	1	-	-
Mitochondria	-	3	4	37
Mitochondria membrane	-	3	-	-
Nucleus	-	15	3	29
Periplasm	3	-	-	-
Plasma membrane	6	5	10	43
Peroxisome	-	-	3	-
Vacuole	-	-	9	5
Vacuole membrane	-	2	-	-
	28	52	43	201

In this table we show the number of sequences per localization in the sequence unique sets of bacterial and eukaryotic SWISS-PROT proteins added between releases 2011_04 and 2012_02 and of *Arabidopsis thaliana* and *Homo sapiens* proteins derived from LocDB (Rastogi and Rost, 2011; Methods). The data sets contained no sequence pairs with HSSP-value>0 and no protein sequences with HSSP-value>5 to any of the sequences used for the development of our prediction method. Note: LocDB annotations of subcellular localization of *A. thaliana* proteins do not discriminate between non-membrane/membrane compartments (with the exception of plasma membrane).

Table SOM_3: Number of proteins in sequence unique sets used for the additional comparison with LocTree

<i>Localization</i>	<i>Bacteria</i>	<i>Eukaryota</i>
Chloroplast	-	11
Cytosol	22	22
Extra-cellular space	20	84
Mitochondria	-	24
Nucleus	-	22
Organelles	-	21
Periplasm	3	-
	45	184

The table displays bacterial and eukaryotic data sets of protein sequences with localization annotations added to Swiss-Prot after 2005. These sets were used for the performance comparison of LocTree2 to LocTree. The sets sequence redundancy reduced internally (HVAL<0 and BLAST2 EVAL \leq 10⁻³ over alignments of \geq 35 residues length; Methods) and to the training sets of LocTree.

Table SOM_4: Evaluation of prediction accuracy and time of various String Kernels

<div>Method</div> <div>Performance</div>		String Subsequence Kernel (Lodhi, et al., 2002)	Mismatch Kernel (Leslie, et al., 2004)	Profile Kernel (Kuang, et al., 2004; SOM Section 2)
<div>Archaea</div> <div>Bacteria</div> <div>Eukaryota</div>	Q(3)	98 ± 3	98 ± 3	98 ± 3
	Q(6)	89 ± 2	89 ± 2	94 ± 2
	Q(18)	70 ± 1	Not available*	83 ± 1

Data set: 10-fold cross-validated training sets of 59 archaeal, 479 bacterial and 1682 eukaryotic proteins (Table SOM_1, cross-validation described in Methods and SOM Section 1).

Performance measures: $Q(n)$, overall prediction accuracy for a given hierarchy (Methods, Qn is a 3-state value for archaea, 6-state value for bacteria and 18-state value for eukaryota); Note: the mismatch kernel was not able to produce results on our largest data sets of eukaryotic proteins within a reasonable amount of time.

The averages over all training sets are reported. The multi-class classification approach used was One-against-all (Allwein, et al., 2000; SOM Section 3).

Table SOM_5: Evaluation of prediction accuracy and time of various multi-class classification techniques

<div> <div>Method</div> <div>Performance</div> </div>		One- against-all (Allwein, et al., 2000)	ENDs (Kramer and Frank, 2004)	ECBNDs (Dong, et al., 2005)	EDBND (Dong, et al., 2005)	Fixed struc- ture (Main pa- per; Fig. 1)
Archaea	Q(3)	98 ± 3	98 ± 3	98 ± 3	98 ± 3	98 ± 3
	Speed	$60 \cdot 10^3$	$6.8 \cdot 10^3$	$8.9 \cdot 10^3$	$8.4 \cdot 10^3$	$40 \cdot 10^3$
Bacteria	Q(6)	94 ± 2	97 ± 2	97 ± 2	97 ± 2	97 ± 2
	Speed	$16.5 \cdot 10^3$	$2 \cdot 10^3$	$2.6 \cdot 10^3$	$2.7 \cdot 10^3$	$15 \cdot 10^3$
Eukaryota	Q(18)	83 ± 1	88 ± 1	88 ± 1	88 ± 1	88 ± 1
	Speed	$4 \cdot 10^3$	$0.2 \cdot 10^3$	$0.3 \cdot 10^3$	$0.3 \cdot 10^3$	$6.1 \cdot 10^3$

Data set: 10-fold cross-validated training sets of 59 archaeal, 479 bacterial and 1682 eukaryotic proteins (Table SOM_1, cross-validation described in Methods and SOM Section 1).

Methods: please refer to SOM Section 3 for methods description.

Performance measures: **Q(n)** used as in Table SOM_3; **Speed**, the number of protein sequences processed per minute on a Dell M605 machine with a Six-Core AMD Opteron processor (2.4 GHz, 6MB and 75W ACP) running on Linux.

The averages over all training sets are reported. The kernel function used was the Profile Kernel (SOM Section_2).

Table SOM_6: LocTree2 on non-redundant test sets of 479 bacterial 1682 and eukaryotic proteins

<i>Localization</i>	<i>Nprot</i>	<i>Acc</i>	<i>Cov</i>	<i>gAv</i>
Cytosol	179	87 ± 6	92 ± 5	89 ± 6
Extra-cellular	82	63 ± 12	73 ± 11	68 ± 11
Fimbrium	16	83 ± 25**	31 ± 32**	51 ± 31
Periplasm	52	75 ± 16	58 ± 16	66 ± 16
Plasma membrane	144	96 ± 4	95 ± 4	95 ± 5
<i>Q(6) – Bacteria</i>		84 ± 4		
Chloroplast	133	44 ± 13	29 ± 9	36 ± 7
Chloroplast membrane	11	38 ± 48**	27 ± 30	32 ± 27
Cytosol	220	45 ± 8	44 ± 8	44 ± 6
ER membrane	65	44 ± 15	42 ± 14	43 ± 11
Extra-cellular	596	80 ± 4	91 ± 3	85 ± 4
Golgi membrane	17	42 ± 33	29 ± 24	35 ± 19
Mitochondria	140	45 ± 10	46 ± 10	45 ± 7
Mitochondria membrane	87	60 ± 15	44 ± 13	51 ± 10
Nucleus	320	67 ± 6	77 ± 6	72 ± 6
Plasma membrane	40	68 ± 22	48 ± 19	57 ± 18
Plastid	14	50 ± 50	21 ± 28	33 ± 21
<i>Q(18) – Eukaryota</i>		65 ± 3		

Abbreviations used: *Nprot*, the number of proteins with known localization; *Acc*, accuracy; *Cov*, coverage; *gAv*, geometric coverage of *Acc* and *Cov*; *Q(n)*, overall prediction accuracy. Standard errors were estimated by bootstrapping (Methods). Note 1: *Q(n)* is a six-state value for bacteria, i.e. the overall accuracy for classification in one of six localization classes, and an eighteen-state value for eukaryota. Note 2: Only performances for localization classes containing more than ten proteins are reported.

** = unrealistic upper or lower bound given by the standard error due to the small data set size.

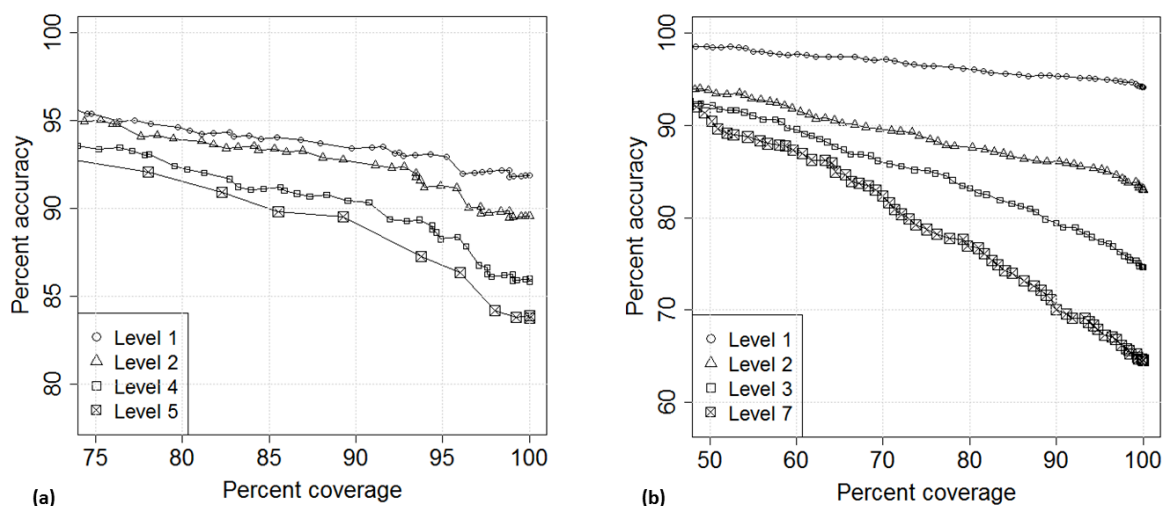
Table SOM_7: LocTree2 on non-redundant test set of 59 archaeal proteins

<i>Localization</i>	<i>Nprot</i>	<i>Acc</i>	<i>Cov</i>	<i>gAv</i>
Cytosol	41	100*	100*	100*
Extra-cellular	5	100*	100*	100*
Plasma membrane	13	100*	100*	100*
Q(3)		100*		

Abbreviations used as for Table SOM_6. Note: $Q(n)$ is a three-state value, i.e. the overall accuracy for classification in one of three localization classes.

* = overoptimistic estimate due to the small data set size.

Figure SOM_1: LocTree2 performance on cross-validated test sets of 479 bacterial and 1682 eukaryotic proteins.



The curves show the overall prediction accuracy on cross-validated sets of sequence unique bacterial and eukaryotic proteins used for the development of LocTree2 (Table SOM_1, Methods). (a) Bacteria: The level 1 accuracy represents the overall two-state accuracy of classifying proteins into cytoplasmic and non-cytoplasmic classes (Fig. 1b). For example, at 90% coverage, the prediction accuracy was around 94%. The overall accuracy declined at lower levels in the hierarchical tree. At Level 2 node that separates proteins into cytosolic, plasma membrane and non-plasma membrane classes, the overall accuracy decreased to 93% at 90% coverage. For the purpose of simplification, the curve for Level 3 predictions is not here provided. Level 4 accuracy includes the accuracies of the cytosolic, plasma membrane, periplasmic space, outer membrane and non-outer membrane classes; it was 91% at 90% coverage. The difference in accuracy between Level 1 and Level 5 predictions that separate proteins in one of six subcellular localization classes was 9%. (b) Eukaryota: the decision node at Level 1 (Fig. 1c) separated membrane spanning proteins from those not associated with membranes. At 80% coverage the accuracy was 96%. Similarly to Fig. 1a, the prediction accuracy declined with the depth of the classification tree. The predictions at Level 2 nodes, where non-membrane and transmembrane proteins are separated into secretory pathway/non-secretory pathway proteins, were made at a lower level of 88% accuracy at 80% coverage. Level 3 nodes separated proteins into eight classes at 83% accuracy and 80% coverage. The prediction into one of eighteen localization classes at Level 7 nodes was performed at a significantly lower accuracy of around 77% at 80% coverage. The performances at Levels 4-6 are explicitly not provided in order to simplify.

Table SOM_8: Performance comparison on LocDB data with several evidences

Method	LocDB (>2 publications)					
	<i>A. thaliana</i> = 10 proteins			<i>H. sapiens</i> = 18 proteins		
	Q(9)	Q(8)	Q(6)	Q(8)	Q(7)	Q(6)
LocTree2	40±34	44±37	60±48	67±27	67±27	85±24
CELLO v. 2.5	40±34	-	-	61±28	-	-
WoLF PSORT	40±34	-	-	61±27	-	-
MultiLoc2	-	22±41	-	-	56±26	-
LOCtree	-	-	60±48	-	-	85±24

Data set: 10 *Arabidopsis thaliana* and 18 *Homo sapiens* sequence-unique proteins from the LocDB database with localization annotations supported by more than two publications. Both data sets were redundancy reduced (HVAL<0; Methods) with respect to each other and to SWISS-PROT 2011_04.

Performance measure: in each column, the highest achieved overall accuracy Q(n) is marked in bold letters; values ± standard error (Methods)

Table SOM_9: Estimating the influence of sequencing mistakes

Method	Full length	30N removed	30C removed	1/3 randomly removed
LocTree2	62 ± 2	54 ± 2	60 ± 2	53 ± 2
CELLO v. 2.5	56 ± 2	35 ± 2	47 ± 2	48 ± 2
WoLF PSORT	56 ± 2	40 ± 2	52 ± 2	49 ± 2

Data set: combined set of all sequence-unique eukaryotic protein sequences extracted from SWISS-PROT and LocDB databases (Tables SOM_1 and SOM_2).

Methods: We estimated and compared the effect of sequencing errors on the performance of LocTree2 and its competitors. Three data sets were used: *30N removed*, the first thirty N-terminal amino acids were cleaved off for all proteins; *30C removed*, the first thirty C-terminal amino acids were cleaved off for all proteins; *1/3 randomly removed*, amino acid positions were randomly picked and cleaved off in-silico until two thirds of the protein sequence remained, which was used to predict localization. *Full length*, is the performance on full length protein sequences and is shown here for comparison.

Performance measure: in each column, the highest achieved overall accuracy is marked in bold letters.

References for Supporting Online Material

- S.F. Altschul, W. Gish, W. Miller, E.W. Myers, D.J. Lipman, *Journal of molecular biology*, 403-410 (1990)
- S.F. Altschul, *et al*, *Nucleic acids research*, 25, 3389-3402. (1997)
- E.L. Allwein, and Y. Singer, *J. Machine Learning Research*, vol. 1, 113-141 (2000)
- A. Bairoch and R. Apweiler, *Nucleic acids research*, **28**, 45-48. (2000)
- Berman, H.M., *et al*. (2000) The Protein Data Bank, *Nucleic acids research*, **28**, 235-242.
- H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H.Weissig, I.N. Shindyalov, P.E. Bourne, *Nucleic acids research*, **28**, 235-242 (2000)
- C. Blake, C.Merz, *University of California, Irvine, Dept. of Inf. and Computer Science* (1998)
[www.ics.uci.edu/~mlearn/MLRepository.html]
- C. Cortes, V. Vapnik, *Machine Learning*, 273-297 (1995)
- L. Dong, E. Frank and S. Kramer, *PKDD* , 84-95 (2005)
- J. Fox, *Applied Regression Analysis, Linear Models, and Related Methods*. Sage Publication. (1997)
- G. Holmes, A. Donkin, I.H. Witten, *Proc Second Australia and New Zealand Conference on Intelligent Information Systems*, 357-361 (1994)
- S. Kramer, E. Frank, *Proceedings of ICML* (2004)
- R. Kuang, E. Ie, K. Wang, K. Wang, M. Siddiqi, Y. Freund, C. Leslie, *Proc IEEE Comput Syst Bioinform Conf*, 152-160 (2004)
- H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, C. Watkins, *The Journal of Machine Learning Research*, vol. 2, 419-444, (2002)
- C. Leslie, E. Eskin, A. Cohen, J. Weston W.S. Noble, *Bioinformatics*, 20(4): 467-476, 2004.
- J.C. Platt, *Advances in Kernel Methods - Support Vector Learning* (1998)
- S. Rastogi, B. Rost, *Nucleic acids research*, D230-234 (2011)
- B. Rost, *Protein engineering*, 85-94 (1999)
- T. Sakurai, T. Moriguchi, K. Furuya, N. Kajiwara, T. Nakamura, M. Yanagisawa, K. Goto, *The Journal of biological chemistry*, **274**(25):17771-17776 (1999)
- C. Sander, R. Schneider, *Proteins*, 56-68 (1991)
- A.M. Villamil Giraldo, M. Lopez Medus, M. Gonzales Lebrero, R.S. Pagano, C.A. Labriola, L. Landolfo, J.M. Delfino, A.J. Parodi, J.J. Caramelo, *The Journal of biological chemistry*, 285(7):4544-4553 (2010)
- G. Webb, X.E. Yu, *Advanced in Artificial Intelligence, Proceedings of 17th Australian Joint Conference on AI, Lecture Notes in Artificial Intelligence* (2004)